Matching across two groups to isolate treatment effects.

Dr. Jon Starkweather, Research and Statistical Support Consultant

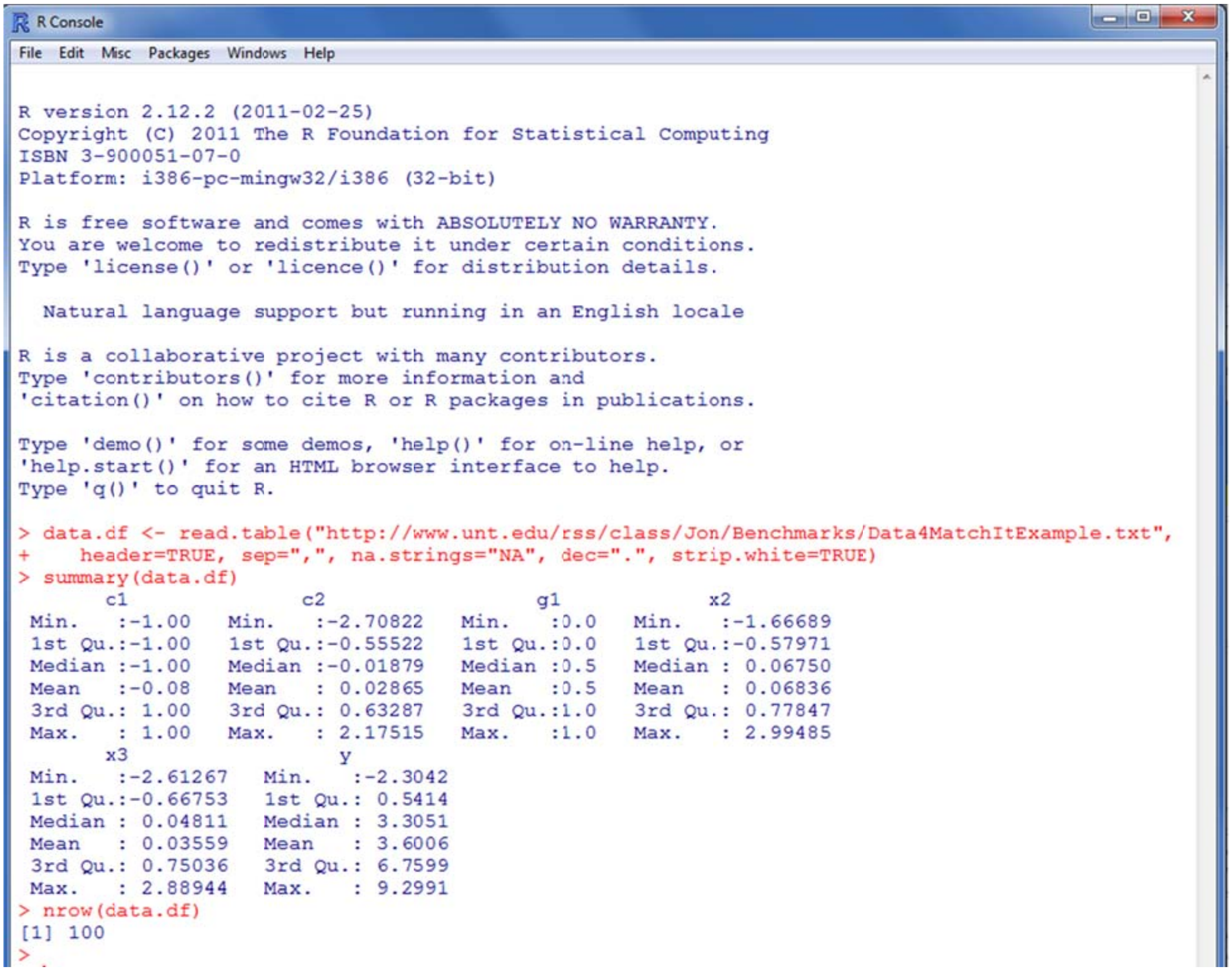
This article will, hopefully, be the first installment of several to discuss the related procedures used to control or remove the influence of *confounder variables*. Here, we define confounder variables as those which have relationships with the primary variables of interest (e.g. moderation, mediation, suppression, etc.). Confounder variables are often identified by the research as being important, but not being of primary interest to the study. Confounder variables are also sometimes called nuisance variables or covariates. Commonly, demographic variables or individual differences (e.g. age, gender, ethnicity, income, etc.) are considered confounders when they are not the primary variables of interest because they so often influence other variables. For example, age may be a meaningful predictor in a linear model with salary as the outcome; while age may be a confounder variable in a model with years of education predicting salary (clearly there is likely to be a meaningful relationship between age and salary). Matching and balancing are virtually the same; for instance, *match*ing cases of the treatment condition with those from the control condition achieves the *balance* one would expect of a truly random sample being truly randomly assigned to the conditions. Clearly then, matching can be used when the design is quasi-experimental; meaning random sampling and / or random assignment are lacking. Practical constraints often lead to this type of design and therefore, the use of matching should be frequently considered. However, matching can also be used when random sampling and random assignment have been carried out, to improve or insure balance among the data.

The 'MatchIt' package (Ho, D., Imai, K., King, G., & Stuart, E., 2011) implements a variety of methods for performing matching across two groups of a predictor based on the values of cases on one or more confounder variables. The resulting balance provides near freedom from some parametric assumptions of many common modeling techniques (e.g. linear regression, general linear model, generalized linear models, hierarchical linear models, structural equation models, etc.). In the regression situation, multicollinearity can be reduced to negligible levels and model specification errors can be controlled; meaning the influence of the confounders on the predictor of interest can be reduced to a point where the direct effect, or main effect, of the dichotomous predictor is independent of confounder influences. As Ho, Imai, King, and Stuart (2007a) state, there are three key advantages to using matching prior to parametric causal modeling; ease of use, more robust parametric estimated parameters – in terms of model form and specification, and reduced bias. The 'MatchIt' functions are easy use as they can be incorporated into typical data analysis routines prior to the primary parametric analysis(es). Parametric estimates based on matched data are more robust to model form and specification errors than raw data parametric estimates because the relationship between the dichotomous predictor variable and the confounder variable(s) has been controlled (i.e. removed or reduced). Reduced bias results from removing the influence of the confounder variables through the matching process; which in turn, decreases the chance of violating the assumptions of some parametric modeling techniques. Ho, et al. (2007a) also reported that the variance of estimated parameters is reduced when using matched data compared to raw data.

The way the ‘matchit’ function (from the ‘MatchIt’ package) works is dependent upon the method of matching used. There are several methods which can be specified by the ‘method’ argument. However, the basic principle of matching is to use a multivariate distance measure (e.g. Mahalanobis distance) to identify cases in the control and treatment groups which responded in the same or similar ways on the confounder variables. Cases which are not matched will be discarded and replaced with replications of cases which were matched. Therefore, sample size remains the same as the original data.

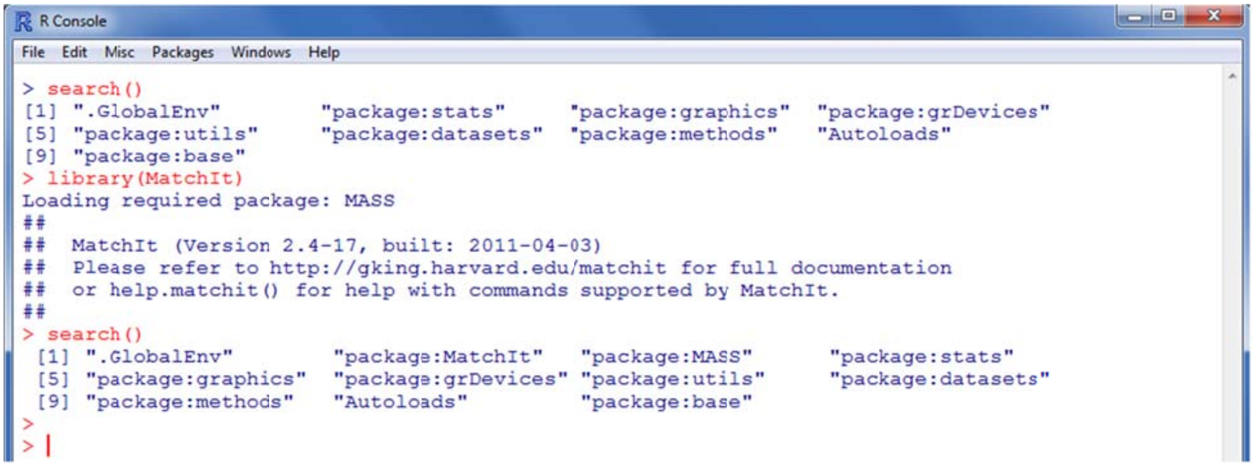
Examples

First, read in the example data from the web naming it “data.df”, get a summary, and take note of the number of rows (nrow). This data is simulated and was created specifically as an example for discussing matching in a regression situation. In the summary output notice that all variables are numeric; although the dichotomous grouping variable (0 = control & 1 = treatment) is g1. The covariates (confounder variables) are c1 and c2; with c1 being dichotomous and c2 being continuous. The continuous outcome variable is y1. Both x2 and x3 are continuous predictors of y1 along with the grouping variable (g1); but x2 and x3 are not related to g1, c1, or c2.



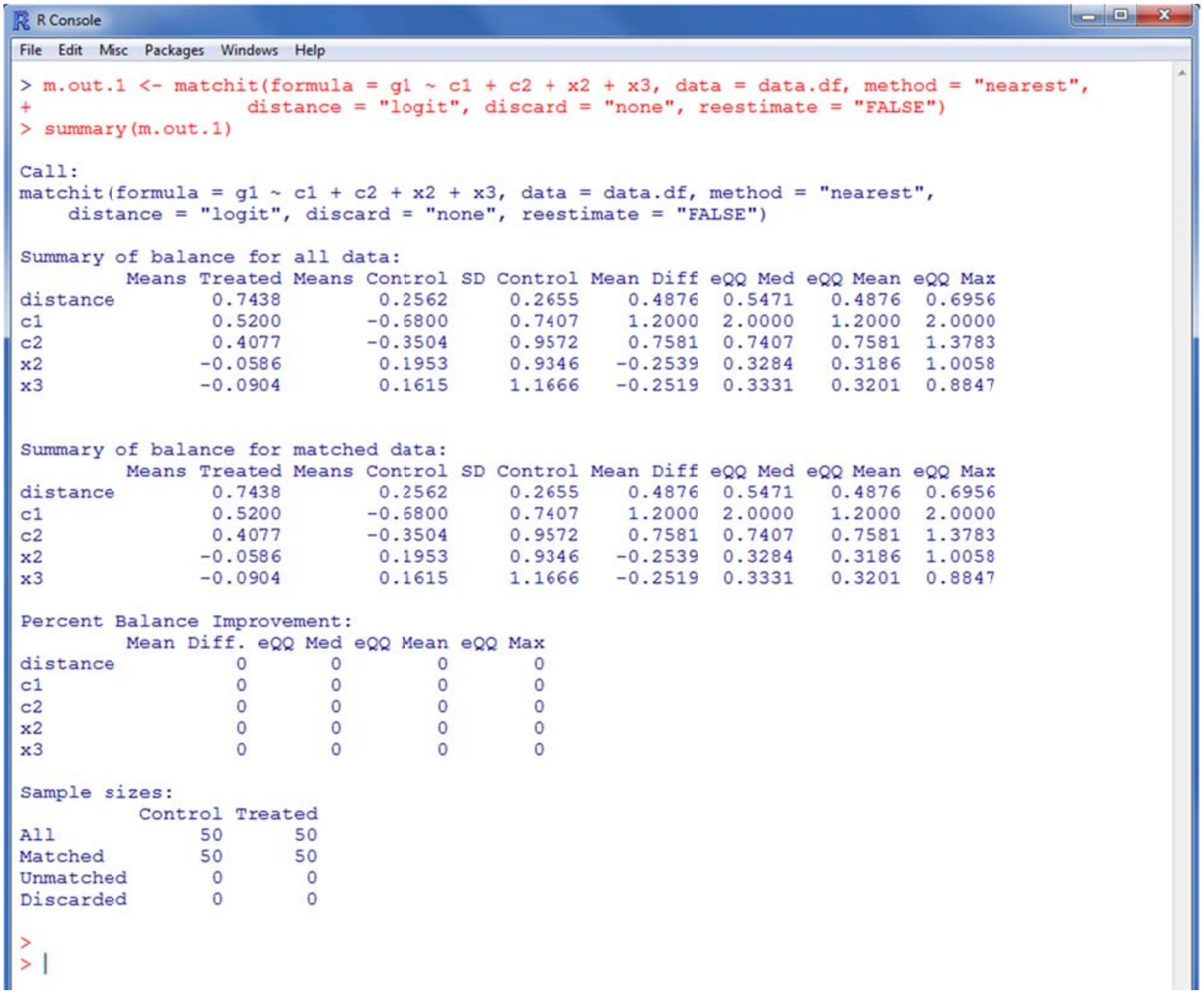
```
R version 2.12.2 (2011-02-25)
Copyright (C) 2011 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: i386-pc-mingw32/i386 (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> data.df <- read.table("http://www.unt.edu/rss/class/Jon/Benchmarks/Data4MatchItExample.txt",
+    header=TRUE, sep=",", na.strings="NA", dec=".", strip.white=TRUE)
> summary(data.df)
       c1               c2                   g1              x2          
 Min.   :-1.00   Min.   :-2.70822   Min.   :0.0   Min.   :-1.66689  
 1st Qu.:-1.00   1st Qu.:-0.55522   1st Qu.:0.0   1st Qu.:-0.57971  
 Median :-1.00   Median :-0.01879   Median :0.5   Median : 0.06750  
 Mean   :-0.08   Mean   : 0.02865   Mean   :0.5   Mean   : 0.06836  
 3rd Qu.: 1.00   3rd Qu.: 0.63287   3rd Qu.:1.0   3rd Qu.: 0.77847  
 Max.   : 1.00   Max.   : 2.17515   Max.   :1.0   Max.   : 2.99485  
       x3                  y          
 Min.   :-2.61267   Min.   :-2.3042  
 1st Qu.:-0.66753   1st Qu.: 0.5414  
 Median : 0.04811   Median : 3.3051  
 Mean   : 0.03559   Mean   : 3.6006  
 3rd Qu.: 0.75036   3rd Qu.: 6.7599  
 Max.   : 2.88944   Max.   : 9.2991  
> nrow(data.df)
[1] 100
>
```

Next, load the ‘MatchIt’ package.



```
> search()
[1] ".GlobalEnv"        "package:stats"     "package:graphics"  "package:grDevices"
[5] "package:utils"     "package:datasets"  "package:methods"   "Autoloads"
[9] "package:base"
> library(MatchIt)
Loading required package: MASS
##
##  MatchIt (Version 2.4-17, built: 2011-04-03)
##  Please refer to http://gking.harvard.edu/matchit for full documentation
##  or help.matchit() for help with commands supported by MatchIt.
##
> search()
 [1] ".GlobalEnv"        "package:MatchIt"   "package:MASS"      "package:stats"
 [5] "package:graphics"  "package:grDevices" "package:utils"     "package:datasets"
 [9] "package:methods"   "Autoloads"         "package:base"
>
>
```

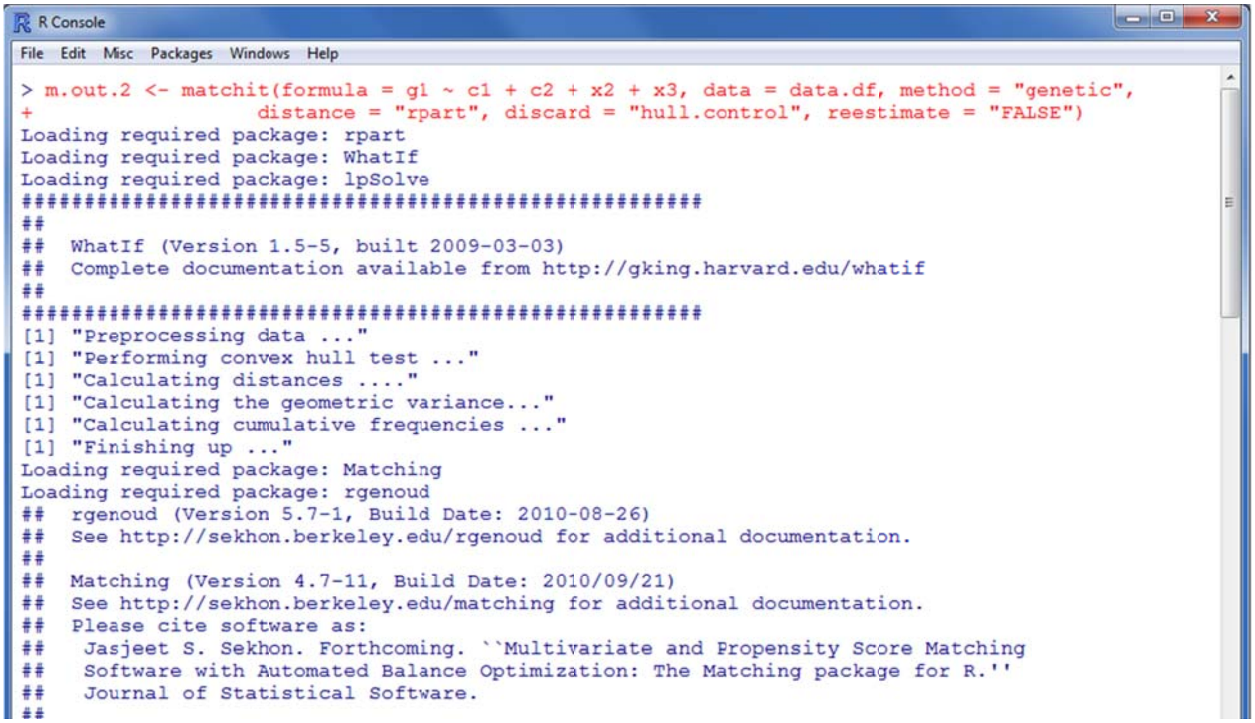
Next, we run our first ‘matchit’ with all the default values for each argument specified and get a summary of the results.



```
> m.out.1 <- matchit(formula = g1 ~ c1 + c2 + x2 + x3, data = data.df, method = "nearest",
+                   distance = "logit", discard = "none", reestimate = "FALSE")
> summary(m.out.1)

Call:
matchit(formula = g1 ~ c1 + c2 + x2 + x3, data = data.df, method = "nearest",
    distance = "logit", discard = "none", reestimate = "FALSE")

Summary of balance for all data:
         Means Treated Means Control SD Control Mean Diff eQQ Med eQQ Mean eQQ Max
distance        0.7438        0.2562     0.2655    0.4876  0.5471   0.4876  0.6956
c1              0.5200       -0.6800     0.7407    1.2000  2.0000   1.2000  2.0000
c2              0.4077       -0.3504     0.9572    0.7581  0.7407   0.7581  1.3783
x2             -0.0586        0.1953     0.9346   -0.2539  0.3284   0.3186  1.0058
x3             -0.0904        0.1615     1.1666   -0.2519  0.3331   0.3201  0.8847


Summary of balance for matched data:
         Means Treated Means Control SD Control Mean Diff eQQ Med eQQ Mean eQQ Max
distance        0.7438        0.2562     0.2655    0.4876  0.5471   0.4876  0.6956
c1              0.5200       -0.6800     0.7407    1.2000  2.0000   1.2000  2.0000
c2              0.4077       -0.3504     0.9572    0.7581  0.7407   0.7581  1.3783
x2             -0.0586        0.1953     0.9346   -0.2539  0.3284   0.3186  1.0058
x3             -0.0904        0.1615     1.1666   -0.2519  0.3331   0.3201  0.8847

Percent Balance Improvement:
         Mean Diff. eQQ Med eQQ Mean eQQ Max
distance          0       0        0       0
c1                0       0        0       0
c2                0       0        0       0
x2                0       0        0       0
x3                0       0        0       0

Sample sizes:
          Control Treated
All            50      50
Matched        50      50
Unmatched       0       0
Discarded       0       0

>
> |
```

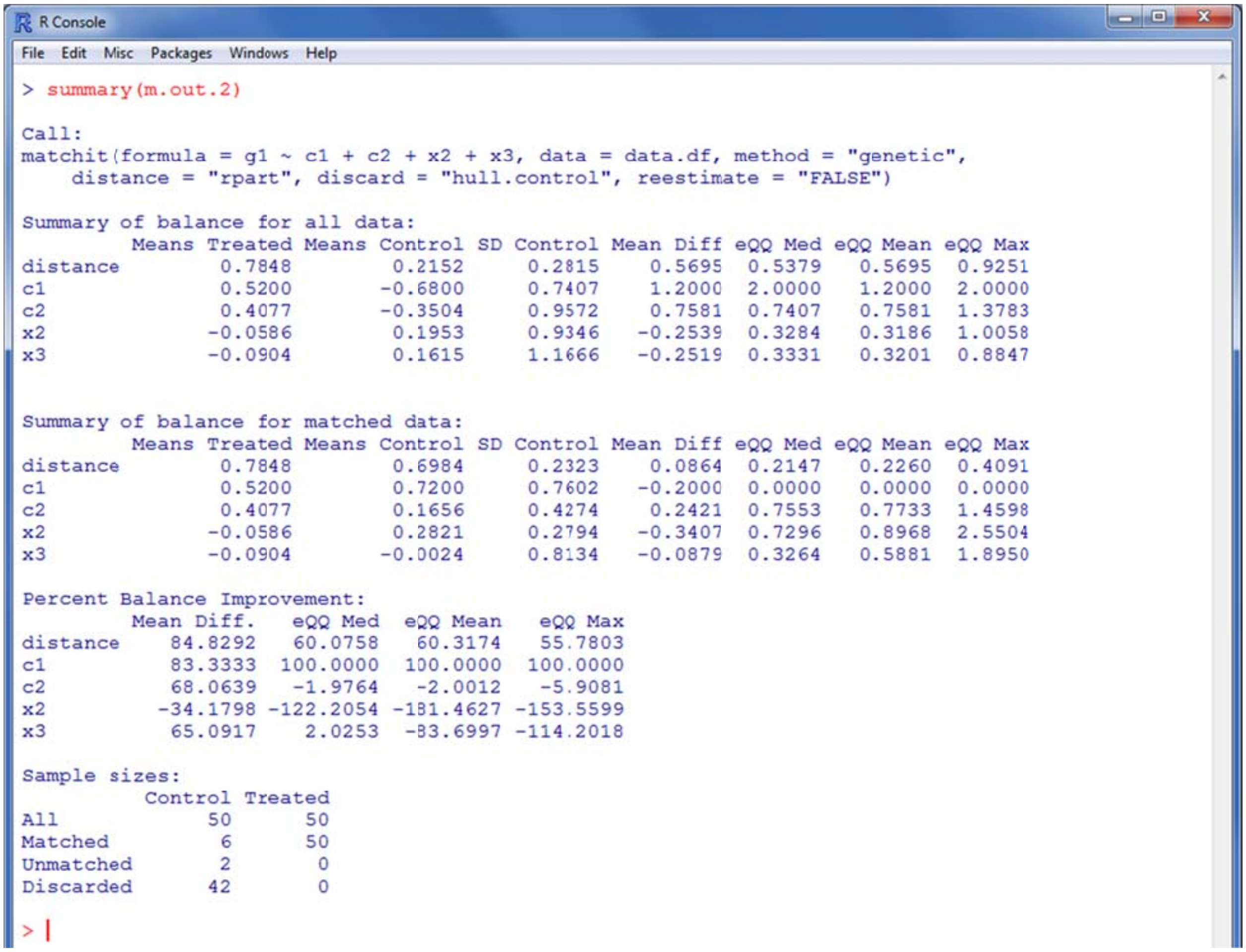
There are four pieces of output produced by the summary function; the summary of balance for "all data" (i.e. the original data), the summary of balance for the "matched data", the percentages of balance improvement, and the sample size summaries. The output of the summary on 'm.out.1' reveals rather strikingly perfect matching. The key elements to focus on are the 'Mean Diff' for the "all data" compared to the 'Mean Diff' for the "matched data" – notice, there were no differences. This is confirmed by noting the Percent Balance Improvement where the zero values indicate a zero percentage change. Furthermore, notice that all the control cases were retained and all the treated cases as well. Essentially, nothing has been done; because, each control case was matched to each treated case on the distance measure; no selection has taken place based on the distance measure. One could use the ‘discard’ optional argument to specify a distance criterion; also called a region of common support which reflects the amount of overlap two variables’ distributions share.

As a comparison; and to show a reason one would want to use the 'matchit' function, we run an example using the "genetic" (algorithm) method, "rpart" distance, and discard "hull.control" which retains all the treatment cases. Notice below, several additional packages are loaded to support the genetic method and optional arguments for it. The ‘Matching’ package (Sekhon, 2009) contains the genetic matching algorithm; function ‘GenMatch’.



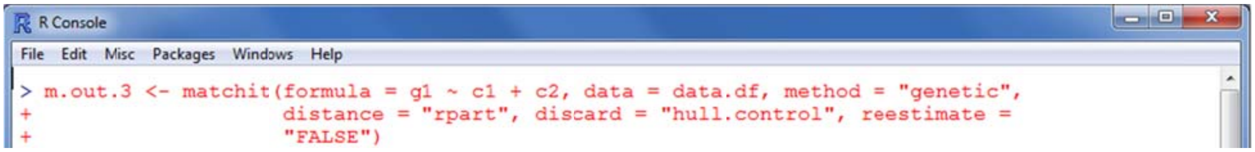
```
> m.out.2 <- matchit(formula = gl ~ c1 + c2 + x2 + x3, data = data.df, method = "genetic",
+                  distance = "rpart", discard = "hull.control", reestimate = "FALSE")
Loading required package: rpart
Loading required package: WhatIf
Loading required package: lpSolve
#######################################################
##
##  WhatIf (Version 1.5-5, built 2009-03-03)
##  Complete documentation available from http://gking.harvard.edu/whatif
##
#######################################################
[1] "Preprocessing data ..."
[1] "Performing convex hull test ..."
[1] "Calculating distances ...."
[1] "Calculating the geometric variance..."
[1] "Calculating cumulative frequencies ..."
[1] "Finishing up ..."
Loading required package: Matching
Loading required package: rgenoud
##  rgenoud (Version 5.7-1, Build Date: 2010-08-26)
##  See http://sekhon.berkeley.edu/rgenoud for additional documentation.
##
##  Matching (Version 4.7-11, Build Date: 2010/09/21)
##  See http://sekhon.berkeley.edu/matching for additional documentation.
##  Please cite software as:
##   Jasjeet S. Sekhon. Forthcoming. ``Multivariate and Propensity Score Matching
##   Software with Automated Balance Optimization: The Matching package for R.''
##   Journal of Statistical Software.
##
```

Iteration history omitted.



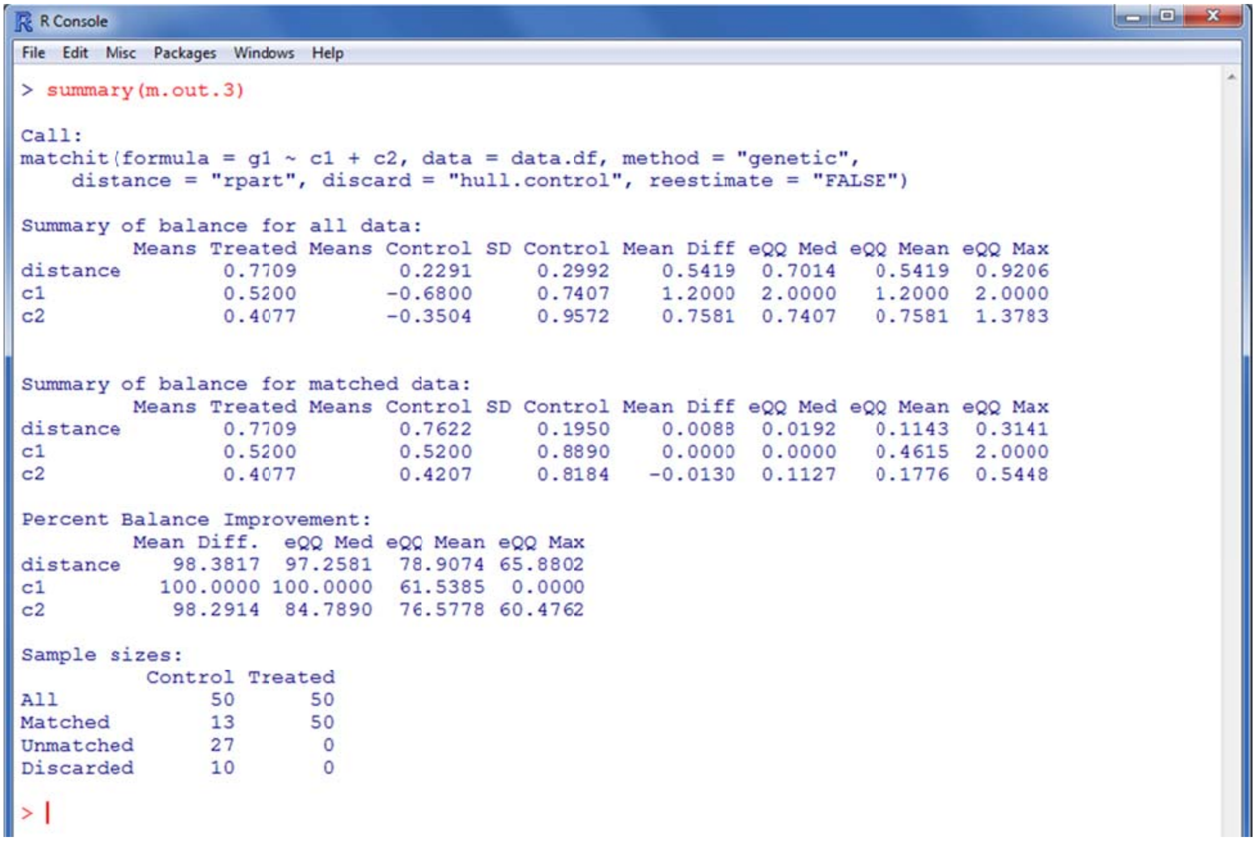
```
> summary(m.out.2)

Call:
matchit(formula = g1 ~ c1 + c2 + x2 + x3, data = data.df, method = "genetic",
    distance = "rpart", discard = "hull.control", reestimate = "FALSE")

Summary of balance for all data:
         Means Treated Means Control SD Control Mean Diff eQQ Med eQQ Mean eQQ Max
distance        0.7848        0.2152     0.2815    0.5695  0.5379   0.5695  0.9251
c1              0.5200       -0.6800     0.7407    1.2000  2.0000   1.2000  2.0000
c2              0.4077       -0.3504     0.9572    0.7581  0.7407   0.7581  1.3783
x2             -0.0586        0.1953     0.9346   -0.2539  0.3284   0.3186  1.0058
x3             -0.0904        0.1615     1.1666   -0.2519  0.3331   0.3201  0.8847


Summary of balance for matched data:
         Means Treated Means Control SD Control Mean Diff eQQ Med eQQ Mean eQQ Max
distance        0.7848        0.6984     0.2323    0.0864  0.2147   0.2260  0.4091
c1              0.5200        0.7200     0.7602   -0.2000  0.0000   0.0000  0.0000
c2              0.4077        0.1656     0.4274    0.2421  0.7553   0.7733  1.4598
x2             -0.0586        0.2821     0.2794   -0.3407  0.7296   0.8968  2.5504
x3             -0.0904       -0.0024     0.8134   -0.0879  0.3264   0.5881  1.8950

Percent Balance Improvement:
         Mean Diff.   eQQ Med  eQQ Mean   eQQ Max
distance    84.8292   60.0758   60.3174   55.7803
c1          83.3333  100.0000  100.0000  100.0000
c2          68.0639   -1.9764   -2.0012   -5.9081
x2         -34.1798 -122.2054 -181.4627 -153.5599
x3          65.0917    2.0253  -83.6997 -114.2018

Sample sizes:
          Control Treated
All            50      50
Matched         6      50
Unmatched       2       0
Discarded      42       0

> |
```

In this summary, we notice that although the mean differences were drastically reduced for the two covariates (c1 & c2), the mean difference actually increased for one of the two predictors (x2). This is a result of those two predictors NOT being related to the grouping variable. So, we might run a third version of the 'matchit' function; including only the two covariates.



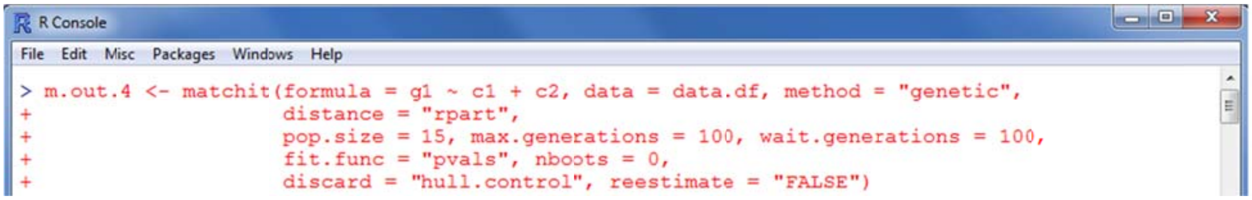
```
> m.out.3 <- matchit(formula = g1 ~ c1 + c2, data = data.df, method = "genetic",
+                    distance = "rpart", discard = "hull.control", reestimate =
+                    "FALSE")
```

Iteration history omitted.



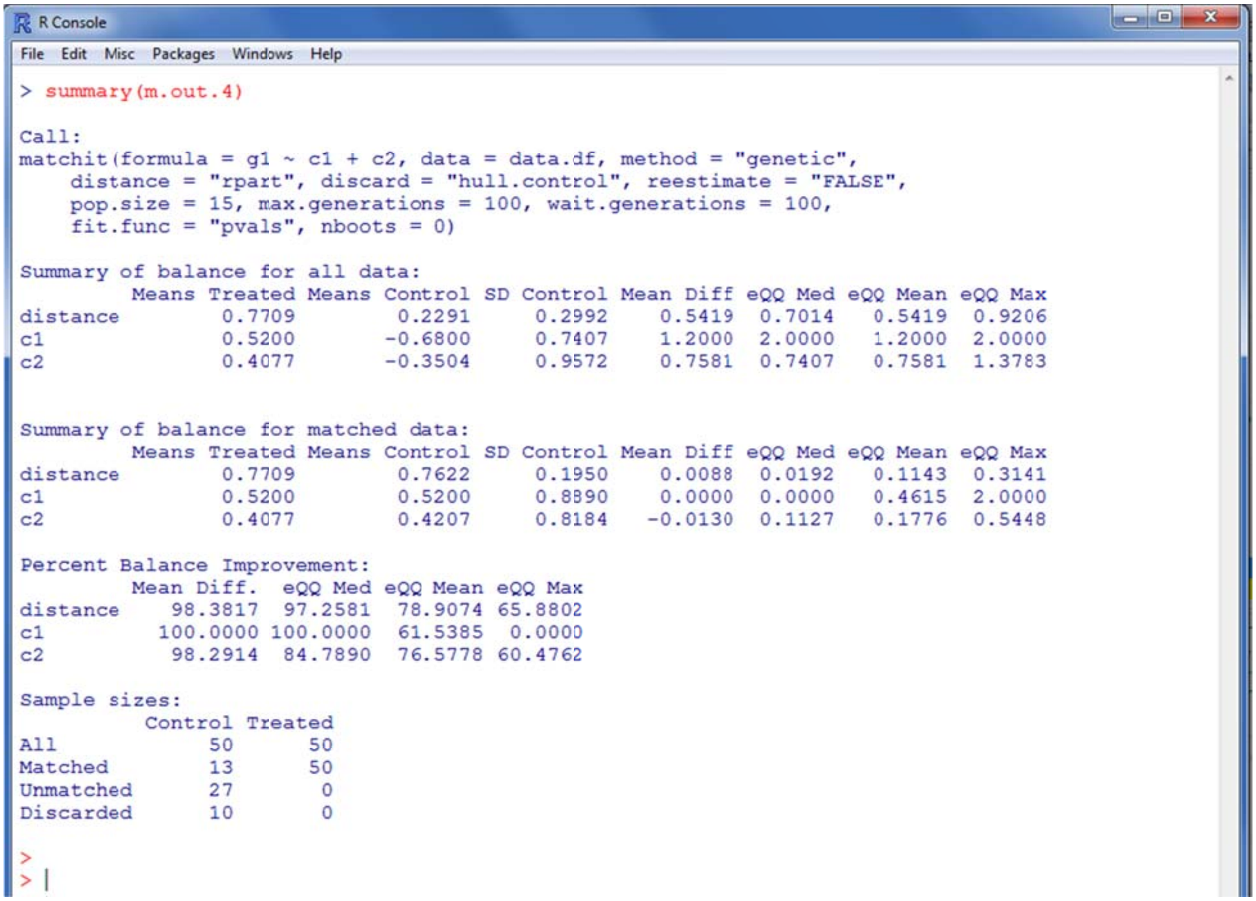
```
> summary(m.out.3)

Call:
matchit(formula = g1 ~ c1 + c2, data = data.df, method = "genetic",
    distance = "rpart", discard = "hull.control", reestimate = "FALSE")

Summary of balance for all data:
         Means Treated Means Control SD Control Mean Diff eQQ Med eQQ Mean eQQ Max
distance        0.7709        0.2291     0.2992    0.5419  0.7014   0.5419  0.9206
c1              0.5200       -0.6800     0.7407    1.2000  2.0000   1.2000  2.0000
c2              0.4077       -0.3504     0.9572    0.7581  0.7407   0.7581  1.3783


Summary of balance for matched data:
         Means Treated Means Control SD Control Mean Diff eQQ Med eQQ Mean eQQ Max
distance        0.7709        0.7622     0.1950    0.0088  0.0192   0.1143  0.3141
c1              0.5200        0.5200     0.8890    0.0000  0.0000   0.4615  2.0000
c2              0.4077        0.4207     0.8184   -0.0130  0.1127   0.1776  0.5448

Percent Balance Improvement:
         Mean Diff.  eQQ Med eQQ Mean eQQ Max
distance    98.3817  97.2581  78.9074 65.8802
c1         100.0000 100.0000  61.5385  0.0000
c2          98.2914  84.7890  76.5778 60.4762

Sample sizes:
          Control Treated
All            50      50
Matched        13      50
Unmatched      27       0
Discarded      10       0

> |
```

With this summary (m.out.3) we see large reductions in the mean differences and the corresponding percent balance improvements. We can also adjust the 'GenMatch' function which is called by the method = "genetic" to better take advantage of the 'GenMatch'; possibly improving the results, but also possibly reducing them. Below, the defaults are shown -- which produce the same output as the previous run (m.out.3). The arguments associated with the ‘GenMatch’ function are pop.size, max.generations, wait.generations, fit.func, and nboots (see Sekhon, 2009).



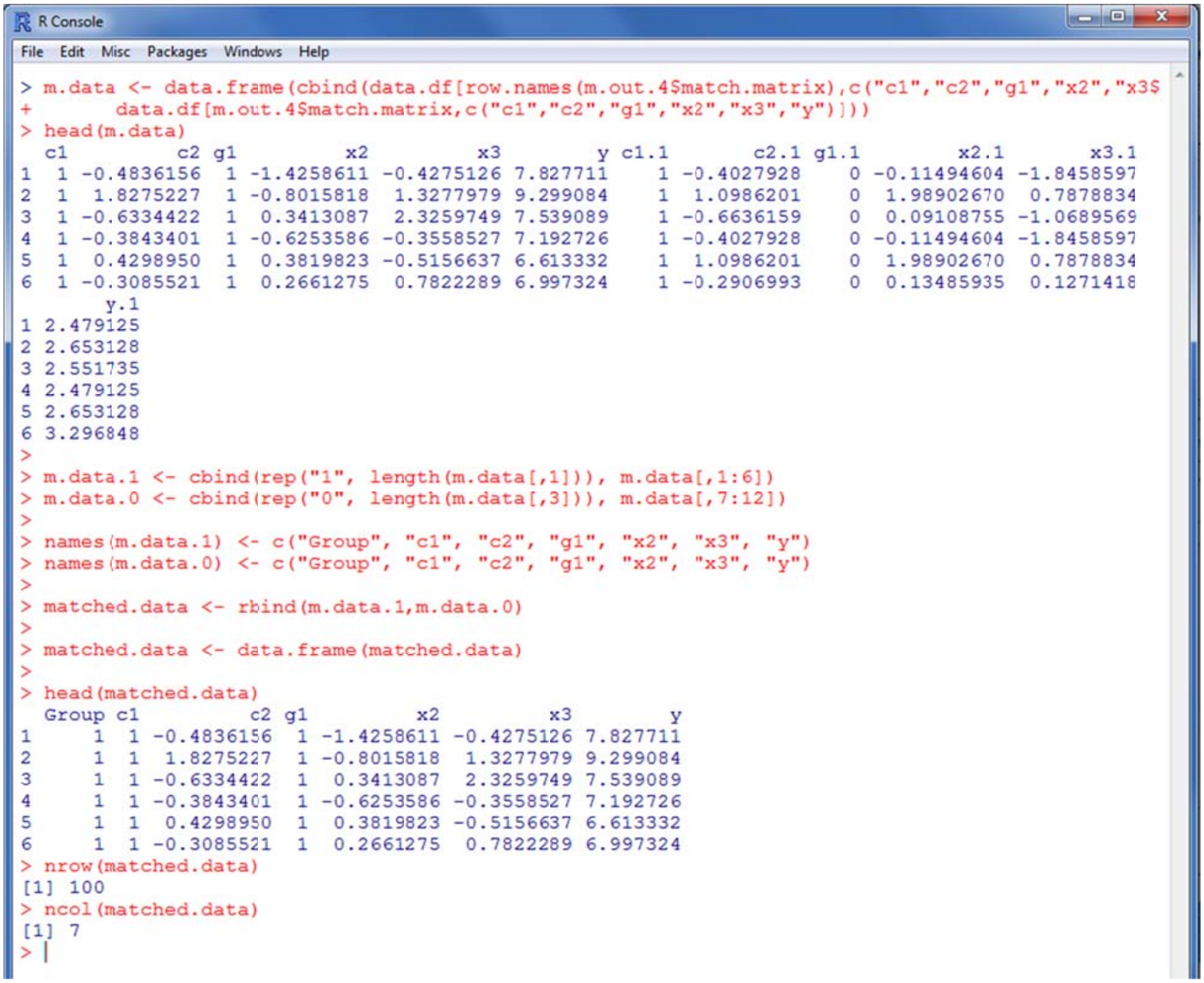
```
> m.out.4 <- matchit(formula = g1 ~ c1 + c2, data = data.df, method = "genetic",
+                    distance = "rpart",
+                    pop.size = 15, max.generations = 100, wait.generations = 100,
+                    fit.func = "pvals", nboots = 0,
+                    discard = "hull.control", reestimate = "FALSE")
```

Iteration history omitted.



```
> summary(m.out.4)

Call:
matchit(formula = g1 ~ c1 + c2, data = data.df, method = "genetic",
    distance = "rpart", discard = "hull.control", reestimate = "FALSE",
    pop.size = 15, max.generations = 100, wait.generations = 100,
    fit.func = "pvals", nboots = 0)

Summary of balance for all data:
         Means Treated Means Control SD Control Mean Diff eQQ Med eQQ Mean eQQ Max
distance        0.7709        0.2291     0.2992    0.5419  0.7014   0.5419  0.9206
c1              0.5200       -0.6800     0.7407    1.2000  2.0000   1.2000  2.0000
c2              0.4077       -0.3504     0.9572    0.7581  0.7407   0.7581  1.3783


Summary of balance for matched data:
         Means Treated Means Control SD Control Mean Diff eQQ Med eQQ Mean eQQ Max
distance        0.7709        0.7622     0.1950    0.0088  0.0192   0.1143  0.3141
c1              0.5200        0.5200     0.8890    0.0000  0.0000   0.4615  2.0000
c2              0.4077        0.4207     0.8184   -0.0130  0.1127   0.1776  0.5448

Percent Balance Improvement:
         Mean Diff.  eQQ Med eQQ Mean eQQ Max
distance    98.3817  97.2581  78.9074 65.8802
c1         100.0000 100.0000  61.5385  0.0000
c2          98.2914  84.7890  76.5778 60.4762

Sample sizes:
          Control Treated
All            50      50
Matched        13      50
Unmatched      27       0
Discarded      10       0

>
> |
```

In order to retrieve or create the new matched data set based on the output from the 'matchit' function, we must do some rather tedious scripting, first, selecting the matched cases by case or row number, then creating a grouping variable to identify each group, then renaming each data frame's columns so they will match when we finally row-bind (rbind) them back together into the ‘match.data’ data frame.

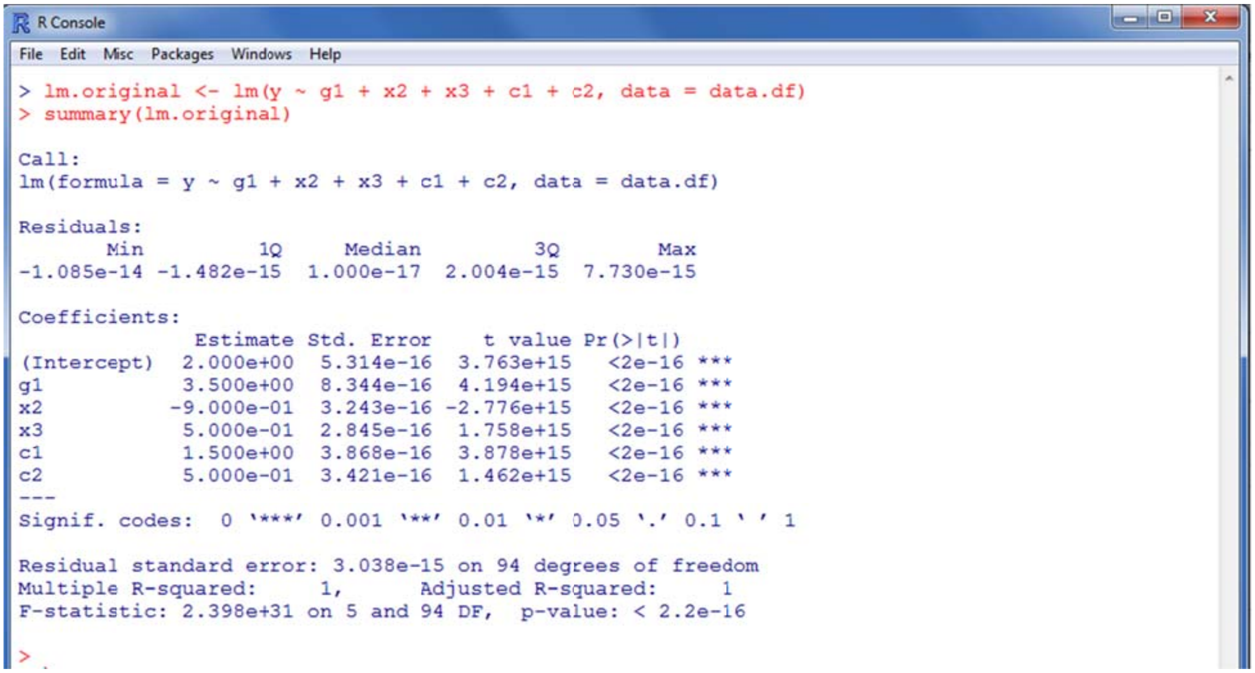


```
> m.data <- data.frame(cbind(data.df[row.names(m.out.4$match.matrix),c("c1","c2","g1","x2","x3$
+       data.df[m.out.4$match.matrix,c("c1","c2","g1","x2","x3","y")]))
> head(m.data)
  c1         c2 g1         x2         x3        y c1.1       c2.1 g1.1        x2.1       x3.1
1  1 -0.4836156  1 -1.4258611 -0.4275126 7.827711    1 -0.4027928    0 -0.11494604 -1.8458597
2  1  1.8275227  1 -0.8015818  1.3277979 9.299084    1  1.0986201    0  1.98902670  0.7878834
3  1 -0.6334422  1  0.3413087  2.3259749 7.539089    1 -0.6636159    0  0.09108755 -1.0689569
4  1 -0.3843401  1 -0.6253586 -0.3558527 7.192726    1 -0.4027928    0 -0.11494604 -1.8458597
5  1  0.4298950  1  0.3819823 -0.5156637 6.613332    1  1.0986201    0  1.98902670  0.7878834
6  1 -0.3085521  1  0.2661275  0.7822289 6.997324    1 -0.2906993    0  0.13485935  0.1271418
       y.1
1 2.479125
2 2.653128
3 2.551735
4 2.479125
5 2.653128
6 3.296848
>
> m.data.1 <- cbind(rep("1", length(m.data[,1])), m.data[,1:6])
> m.data.0 <- cbind(rep("0", length(m.data[,3])), m.data[,7:12])
>
> names(m.data.1) <- c("Group", "c1", "c2", "g1", "x2", "x3", "y")
> names(m.data.0) <- c("Group", "c1", "c2", "g1", "x2", "x3", "y")
>
> matched.data <- rbind(m.data.1,m.data.0)
>
> matched.data <- data.frame(matched.data)
>
> head(matched.data)
  Group c1         c2 g1         x2         x3        y
1     1  1 -0.4836156  1 -1.4258611 -0.4275126 7.827711
2     1  1  1.8275227  1 -0.8015818  1.3277979 9.299084
3     1  1 -0.6334422  1  0.3413087  2.3259749 7.539089
4     1  1 -0.3843401  1 -0.6253586 -0.3558527 7.192726
5     1  1  0.4298950  1  0.3819823 -0.5156637 6.613332
6     1  1 -0.3085521  1  0.2661275  0.7822289 6.997324
> nrow(matched.data)
[1] 100
> ncol(matched.data)
[1] 7
> |
```

Keep in mind, this new (matched) data can be saved or written out of R using the standard functions (e.g. write.table, write.csv, etc.).

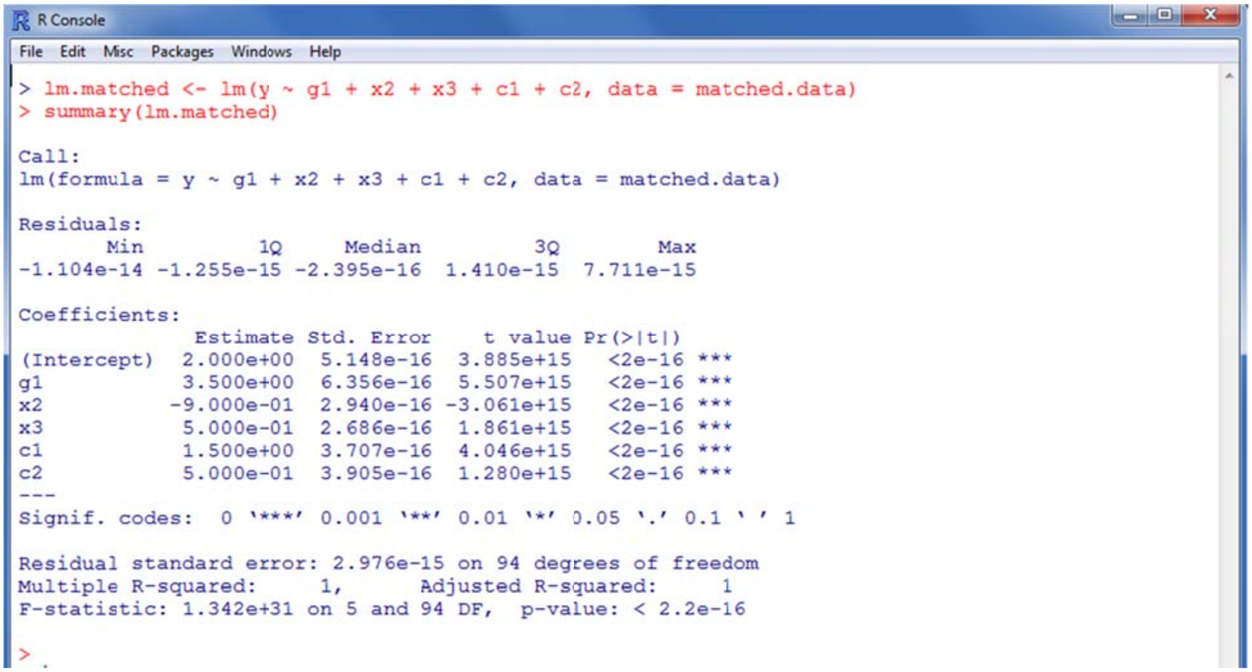
Now we can do some comparisons to see how the matching has affected our analysis of the data; in terms of the proposed model we will now test. Here we are using a simple linear model, but keep in mind the model could be a complex SEM or HLM or whatever. In this example; we use linear regression. Keep in mind, the data is simulated and was generated with massive effects (i.e. no measurement error) and therefore, the coefficients are exactly modeled.

First, we run the linear model with the original (non-matched) data.



```
> lm.original <- lm(y ~ g1 + x2 + x3 + c1 + c2, data = data.df)
> summary(lm.original)

Call:
lm(formula = y ~ g1 + x2 + x3 + c1 + c2, data = data.df)

Residuals:
       Min         1Q     Median         3Q        Max
-1.085e-14 -1.482e-15  1.000e-17  2.004e-15  7.730e-15

Coefficients:
              Estimate Std. Error    t value Pr(>|t|)
(Intercept)  2.000e+00  5.314e-16  3.763e+15   <2e-16 ***
g1           3.500e+00  8.344e-16  4.194e+15   <2e-16 ***
x2          -9.000e-01  3.243e-16 -2.776e+15   <2e-16 ***
x3           5.000e-01  2.845e-16  1.758e+15   <2e-16 ***
c1           1.500e+00  3.868e-16  3.878e+15   <2e-16 ***
c2           5.000e-01  3.421e-16  1.462e+15   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.038e-15 on 94 degrees of freedom
Multiple R-squared:     1,      Adjusted R-squared:     1
F-statistic: 2.398e+31 on 5 and 94 DF,  p-value: < 2.2e-16

>
```

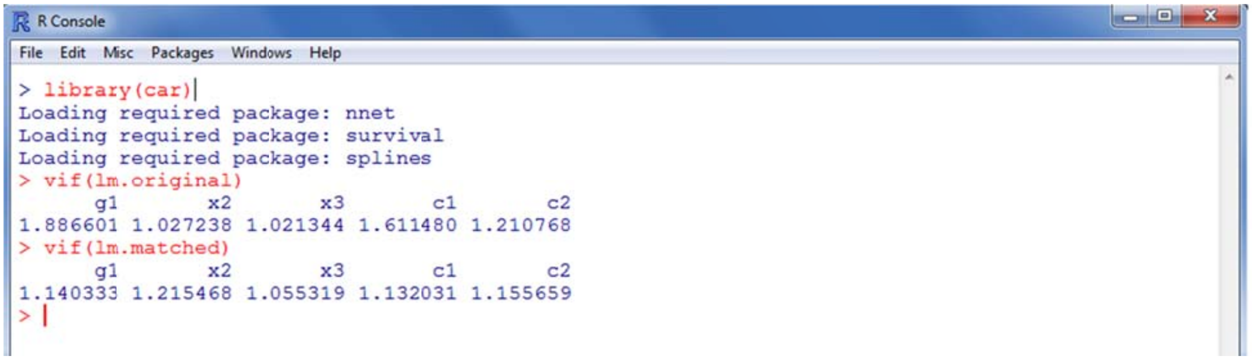
Second, we run the linear model with the matched data.



```
> lm.matched <- lm(y ~ g1 + x2 + x3 + c1 + c2, data = matched.data)
> summary(lm.matched)

Call:
lm(formula = y ~ g1 + x2 + x3 + c1 + c2, data = matched.data)

Residuals:
       Min         1Q     Median         3Q        Max
-1.104e-14 -1.255e-15 -2.395e-16  1.410e-15  7.711e-15

Coefficients:
              Estimate Std. Error    t value Pr(>|t|)
(Intercept)  2.000e+00  5.148e-16  3.885e+15   <2e-16 ***
g1           3.500e+00  6.356e-16  5.507e+15   <2e-16 ***
x2          -9.000e-01  2.940e-16 -3.061e+15   <2e-16 ***
x3           5.000e-01  2.686e-16  1.861e+15   <2e-16 ***
c1           1.500e+00  3.707e-16  4.046e+15   <2e-16 ***
c2           5.000e-01  3.905e-16  1.280e+15   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.976e-15 on 94 degrees of freedom
Multiple R-squared:     1,      Adjusted R-squared:     1
F-statistic: 1.342e+31 on 5 and 94 DF,  p-value: < 2.2e-16

>
```

It may be a bit odd that the output of each of the linear models is virtually the same. That is because the simulated data was created in such a way as to have extremely large effect sizes. However, if we look closely at the standard errors and the t-values (of the variables involved in the matching) we can see that the linear model with the matched data is more accurately capturing the main effects of each of those variables; because, we have decreased the strength of the relationships between the grouping variable and the covariates.

We can also document improvement by taking a look at the Variance Inflation Factor (VIF) for the grouping variable and both covariates. Notice, the VIFs for the matched variables (g1, c1, & c2) are notably lower than in the original data.



```
> library(car)
Loading required package: nnet
Loading required package: survival
Loading required package: splines
> vif(lm.original)
      g1       x2       x3       c1       c2 
1.886601 1.027238 1.021344 1.611480 1.210768 
> vif(lm.matched)
      g1       x2       x3       c1       c2 
1.140333 1.215468 1.055319 1.132031 1.155659 
> 
```

Conclusion

The ‘matchit’ function in the ‘MatchIt’ package can be used for balancing the effect of one or more confounder variables (covariates) across a dichotomous grouping variable when in a variety of modeling situations. The function works on all types of covariates; be they dichotomous, polytomous, or continuous. In the linear modeling situation, the outcome variable can be of any type as well. The key to matching with ‘matchit’ is that the outcome is not used by the function. The function only deals with the relationships between a dichotomous predictor variable and other predictor variables included in the modeling strategy; be they variables of interest or covariates. These relationships, multicollinearity, cause indirect effects on the outcome which degrade the validity of interpretations and conclusions based on the coefficients of the predictors of interest and/or the covariates. In essence, the presence of indirect effects *confounds* the validity of the modeled direct effects. The direct effects are represented by the individual predictor coefficients or parameters. Indirect effects, multicollinearity being one of them, can manifest in a variety of ways; such as suppression. Suppression not only degrades the accuracy of the coefficients, but can cause the sign of a coefficient to be reversed as in the case of Simpson’s paradox. By not taking into account the relationships among the predictors (of interest or covariates) the beta coefficients returned will be inaccurate representations of the direct effects of their respective variables on the outcome variable. In the presence of Simpson’s paradox, the inaccuracy would include a change in sign. For example, a predictor may have a negative coefficient when taking into account the relationship between it and another predictor but, may display a positive coefficient when not taking that relationship into account. Matching balances those types of indirect effects across the groups of a dichotomous variable so that the direct effects of all the predictors are accurately modeled.

An Adobe.pdf version of this article can be found here.

Until next time; *everything is made of dreams…*

References & Resources

Ho, D., Imai, K., King, G., & Stuart, E. (2007a). Matching as nonparametric preprocessing for reducing model dependence in parametric causal inference. *Political Analysis*, *15*(3), 199 – 236.

Ho, D., Imai, K., King, G., & Stuart, E. (2007b). Matchit: Nonparametric preprocessing for parametric causal inference. *Journal of Statistical Software;* http://gking.harvard.edu/matchit/

Ho, D., Imai, K., King, G., & Stuart, E. (2011). Package 'MatchIt'. Available at: http://cran.r-project.org/web/packages/MatchIt/index.html

Sekhon, J. (2009). Package 'Matching'. Available at: http://cran.r-project.org/web/packages/Matching/index.html

Waits, T. (1998). *Temptation.* From the album: Beautiful Maladies.